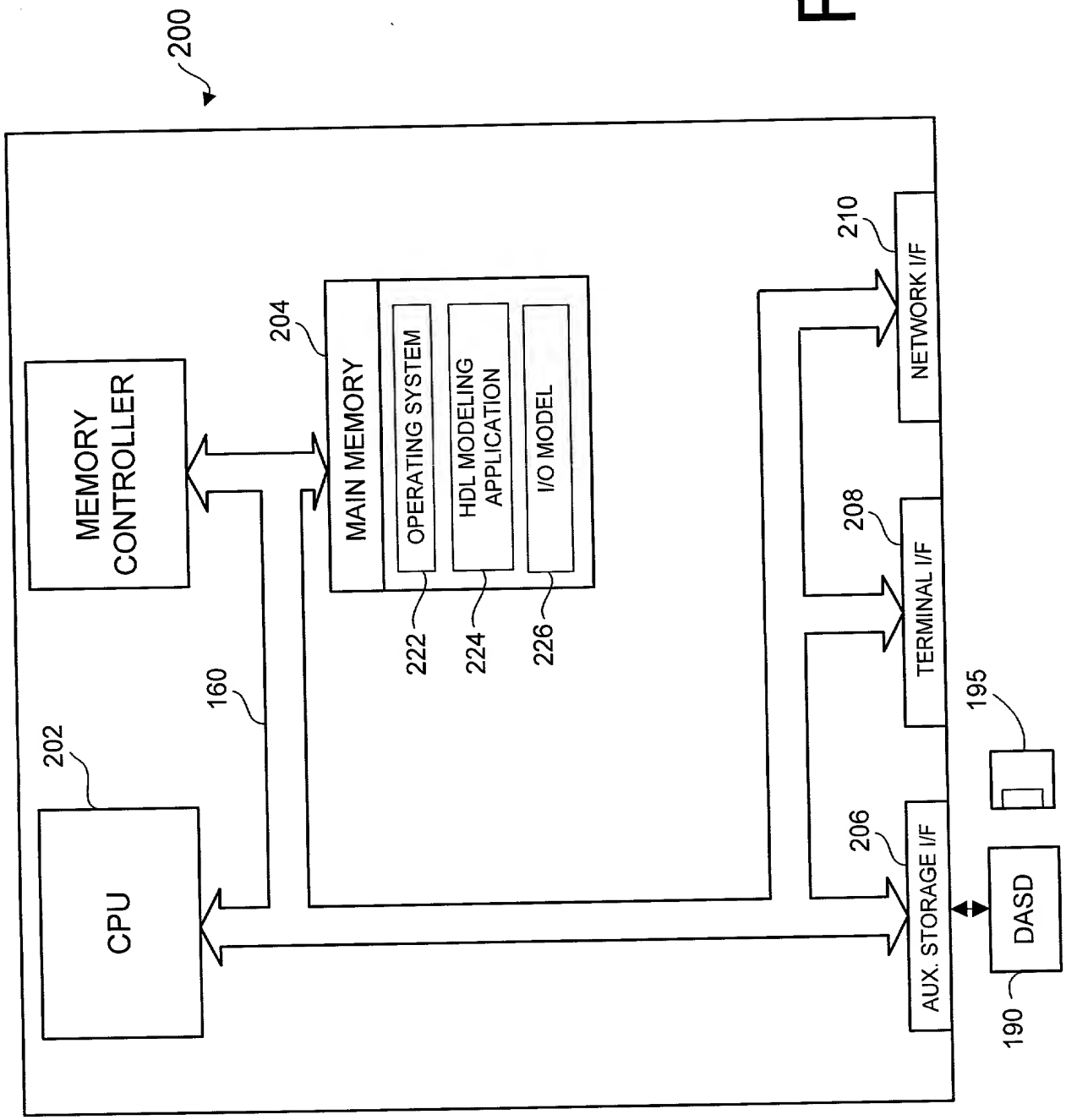
200
MEMORY CONTROLLER
204
MAIN MEMORY
OPERATING SYSTEM
HDL MODELING APPLICATION
I/O MODEL
222
224
226
210
NETWORK I/F
208
TERMINAL I/F
195
160
202
CPU
206
AUX. STORAGE I/F
DASD
190

FIG. 1

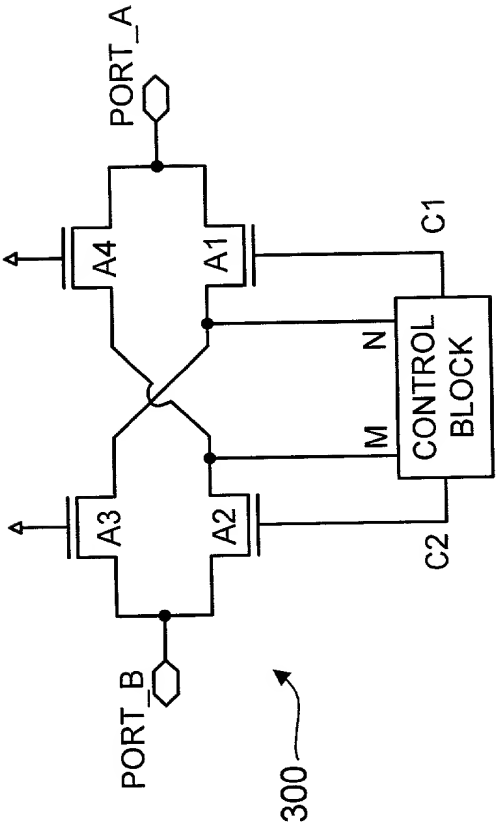

FIG. 2

VERILOG EXAMPLE

```
module BiDiWireIO (PORT_A,  PORT_B) ;
inout  PORT_A,  PORT_B;
reg  cl,  c2;

initial begin
c1=1 'b0;
c2=1 'b0;
end

nmos  A1     (PORT_A,  n,  c1) ;
nmos  A3     (n,  PORT_B,  1 'b1) ;

nmos  A2     (PORT_B,  m,  c2) ;
nmos  A4     (m,  PORT_A,  1 'b1) ;

always  @ (m)
if  (m  ! = =  n)  begin
   c1  =  0;
   c2  =  1;
end

always  @ (n)
if  (m  ! = =  n)  begin
   c1  =  1;
   c2  =  0;
end

specify
   (PORT_B  +=>  PORT_A)  =  (0 . 0 : 0 . 0 : 0 . 0  ,  0 . 0 : 0 . 0 : 0 . 0) ;
   (PORT_A  +=>  PORT_B)  =  (0 . 0 : 0 . 0 : 0 . 0  ,  0 . 0 : 0 . 0 : 0 . 0) ;
endspecify

endmodule
```

FIG. 3

VERILOG NMOS PRIMITIVE
TRUTH TABLE

INPUT \ CONTROL	0	1	X	Z
0	Z	0	L	L
1	Z	1	H	H
X	Z	X	X	X
Z	Z	Z	Z	Z

400

FIG. 4

VITAL HDL EXAMPLE (PART 1)

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
library IEEE;
use IEEE.VITAL_Timing.all;

-- entity declaration --
entity BiDiWireIO is
    generic (
        tpd_PORT_B_PORT_A                : VitalDelayType01 := (0.100 ns, 0.100 ns);
        tpd_PORT_A_PORT_B                : VitalDelayType01 := (0.100 ns, 0.100 ns);
        tipd_PORT_A                      : VitalDelayType01 := (0.100 ns, 0.100 ns);
        tipd_PORT_B                      : VitalDelayType01 := (0.100 ns, 0.100 ns);

    port (
        PORT_B                           : inout STD_LOGIC;
        PORT_A                           : inout STD_LOGIC);
attribute VITAL_LEVEL0 of BiDiWireIO : entity is TRUE;
end BiDiWireIO;

-- architecture body --
library IEEE;
use IEEE.VITAL_Primitives.all;
architecture ARCH_VITAL of BiDiWireIO is

        SIGNAL PORT_B_ipd          : STD_LOGIC := 'Z';
        SIGNAL PORT_A_ipd          : STD_LOGIC := 'Z';


        SIGNAL PORT_B_zd           : STD_LOGIC := 'Z';
        SIGNAL PORT_A_zd           : STD_LOGIC := 'Z';

        SIGNAL c1                  : STD_LOGIC := '0';
        SIGNAL c2                  : STD_LOGIC := '0';

begin

        WireDelay : block
        begin
        VitalWireDelay (PORT_A_ipd, PORT_A, tipd_PORT_A);
        VitalWireDelay (PORT_B_ipd, PORT_B, tipd_PORT_B);
        end block;
```

FIG. 5

VITAL HDL EXAMPLE (PART 2)

```
PORT_A_zd <= PORT_B_ipd   when  c1 = '1' else 'Z';
PORT_B_zd  <= PORT_A_ipd when  c2 = '1' else 'Z';

c1 <= '1'  when PORT_B 'event   and (PORT_A /= PORT_B) else
      '0'  when PORT_A 'event and (PORT_A /= PORT_B) else
      c1;

c2  <= '0'  when PORT_B 'event   and (PORT_A /= PORT_B) else
       '1'  when PORT_A 'event and (PORT_A /= PORT_B) else
       c2;

VITALBehavior  :  process  (PORT_A_zd,  PORT_B_zd)

    variable  PORT_A_GlitchData       : VitalGlitchDataType;
    variable  PORT_B_GlitchData       : VitalGlitchDataType;

begin
       VitalPathDelay01  (
        OutSignal  =>  PORT_A,
        GlitchData  =>  PORT_A_GlitchData,
        OutSignalName  =>  "PORT_A",
        OutTemp  =>  PORT_A_zd,
        Paths  =>  (0  =>  (PORT_B_ipd 'last_event,
tpd_PORT_B_PORT_A,  TRUE)),
       Mode => OnDetect,
       Xon => True,
       MsgOn => True,
       MsgSeverity => WARNING) ;

       VitalPathDelay01  (
        OutSignal  =>  PORT_B,
        GlitchData  =>  PORT_B_GlitchData,
        OutSignalName  =>  "PORT_B",
        OutTemp  =>  PORT_B_zd,
        Paths  =>  (0  =>  (PORT_A_ipd 'last_event,
tpd_PORT_A_PORT_B,  TRUE)),
       Mode => OnDetect,
       Xon => True,
       MsgOn => True,
       MsgSeverity => WARNING) ;

end process;
end ARCH_VITAL
```

FIG. 6